IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants:	David E. Lowell, et al.	Examiner: Meng Yao Zhe
Serial No.:	10/676,665	Group Art Unit: 2195
Filed:	October 1, 2003	Docket No.: 200208636-1
Title:	Online Computer Maintenance Utilizing a Virtual Machine Monitor	

APPEAL BRIEF UNDER 37 C.F.R. § 41.37

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This Appeal Brief is filed in response to the Final Office Action mailed November 16, 2007 and Notice of Appeal filed on February 19, 2008.

AUTHORIZATION TO DEBIT ACCOUNT

It is believed that no extensions of time or fees are required, beyond those that may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, such extensions are hereby petitioned under 37 C.F.R. § 1.136(a), and any fees required (including fees for net addition of claims) are hereby authorized to be charged to Hewlett-Packard Development Company's deposit account no. 08-2025.

I. REAL PARTY IN INTEREST

The real party in interest is Hewlett-Packard Development Company, LP, a limited partnership established under the laws of the State of Texas and having a principal place of business at 20555 S.H. 249 Houston, TX 77070, U.S.A. (hereinafter "HPDC"). HPDC is a Texas limited partnership and is a wholly-owned affiliate of Hewlett-Packard Company, a Delaware Corporation, headquartered in Palo Alto, CA. The general or managing partner of HPDC is HPQ Holdings, LLC.

II. RELATED APPEALS AND INTERFERENCES

There are no known related appeals, judicial proceedings, or interferences known to appellant, the appellant's legal representative, or assignee that will directly affect or be directly affected by or have a bearing on the Appeal Board's decision in the pending appeal.

III. STATUS OF CLAIMS

Claims 1 – 43 are pending in the application and stand finally rejected. The rejection of claims 1 – 43 is appealed.

IV. STATUS OF AMENDMENTS

No amendments were made after receipt of the Final Office Action. All amendments have been entered.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The following provides a concise explanation of the subject matter defined in each of the claims involved in the appeal, referring to the specification by page and line number and to the drawings by reference characters, as required by 37 C.F.R. § 41.37(c)(1)(v). Each element of the claims is identified by a corresponding reference to the specification and drawings where applicable. Note that the citation to passages in the specification and drawings for each claim element does not imply that the limitations from the specification and drawings should be read into the corresponding claim element or that these are the sole sources in the specification supporting the claim features.

Claim 1

A method of performing online computer maintenance on at least one node, the method comprising:

running a virtual machine monitor (Fig. 2, #210: a virtual machine monitor (VMM) is runs on a node: see p. 4, lines 1-3 of paragraph [0017]);

running a first operating system instance on the virtual machine monitor (Fig. 2, #212:one or more operating system (OS) instances are run on the VMM: see p. 4, lines 1-2 of paragraph [0018]);

configuring hardware to trap instructions executed by the first operating system instance (when the VMM starts an OS instance, it configures the hardware to trap when the OS instance executes privileged instructions: see p. 4, lines 3-5 of paragraph [0018]);

simulating trapped instructions with the virtual machine monitor (when a trap occurs, the VMM simulates the instruction and thus creates an illusion that the OS instance has sole control of the hardware on which it runs: see p. 4, line 5 of paragraph [0018] to p. 5, lines 1-2 of paragraph [0018]);

running a second operating system instance on the virtual machine monitor as a substitute for the first operating system instance (Fig. 2, #216: a second OS is run as a substitute for the first OS instance and proves the same services as the first OS instance by taking over the services temporarily or permanently: see p. 5, lines 1-4 of paragraph [0021]); and

performing the maintenance with respect to one of the first and second operating system instances while using the other of the first and second operating system instances (Fig. 2, #218: maintenance is performed with respect to one of the first and second OS instances while using the other of those instances. For example, maintenance is performed on the first OS instance while using the second OS instance: see p. 5, lines 1-5 of paragraph [0022]).

Claim 2

The method of claim 1, wherein the second operating system instance is run as a substitute by migrating at least one application from the first operating system instance to the second operating system instance, and using the migrated applications on the second operating system instance (applications are migrated from the first OS instance to the second OS instance: see p. 6, lines 1-4 of paragraph [0024]).

Claim 21

A node (Fig. 1, #110: p. 4, line 1 of paragraph [0016]) comprising a processing unit (Fig. 1, #114: p. 4, line 2 of paragraph [0016]) and memory (Fig. 1, #116: p. 4, line 2 of paragraph [0016]) for the processing unit, the memory encoded with a virtual machine monitor and an operating system (the memory stores a virtual machine monitor application and operating systems: see p. 4, lines 4-6 of paragraph [0016]), the virtual machine monitor running a second instance of the operating system when a first instance of the operating system is already running in the node (Fig. 2, #212:one or more operating system (OS) instances are run on the VMM: see p. 4, lines 1-2 of paragraph [0018]), the second instance being run when maintenance is to be performed(Fig. 2, #218: maintenance is performed with respect to one of the first and second OS instances while using the other of those instances. For example, maintenance is performed on the first OS instance while using the second OS instance: see p. 5, lines 1-5 of paragraph [0022]), the second instance being a substitute for the first instance (Fig. 2, #216: a second OS is run as a substitute for the first OS instance and proves the same services as the first OS instance by taking over the services temporarily or permanently: see p. 5, lines 1-4 of paragraph [0021]), the virtual machine monitor allowing the maintenance to

be performed with respect to one of the instances while using the other of the instances wherein the virtual machine monitor configures hardware to trap privileged instructions to create an illusion that the first instance has control of the hardware (when a trap occurs, the VMM simulates the instruction and thus creates an illusion that the OS instance has sole control of the hardware on which it runs: see p. 4, line 5 of paragraph [0018] to p. 5, lines 1-2 of paragraph [0018]).

Claim 22

The node of claim 21, wherein the second instance is run as a substitute by migrating at least one application from the first instance to the second instance, and using the at least one application on the second instance (applications are migrated from the first OS instance to the second OS instance: see p. 6, lines 1-4 of paragraph [0024]).

Claim 33

An article (Fig. 1, #112 shows a computer: p. 4, line 2 of paragraph [0016]) for a processing unit (Fig. 1, #114: p. 4, line 2 of paragraph [0016]) of a node (Fig. 1, #110: p. 4, line 1 of paragraph [0016]), the article comprising computer memory (Fig. 1, #116: p. 4, line 2 of paragraph [0016]) encoded with a virtual machine monitor for running first and second instances of an operating system (the memory stores a virtual machine monitor application and operating systems: see p. 4, lines 4-6 of paragraph [0016]), the second instance being run when maintenance on the node is to be performed (Fig. 2, #218: maintenance is performed with respect to one of the first and second OS instances while using the other of those instances: see p. 5, lines 1-5 of paragraph [0022]), the second instance being a substitute for the first instance (Fig. 2, #216: a second OS is run as a substitute for the first OS instance and proves the same services as the first OS instance by taking over the services temporarily or permanently: see p. 5, lines 1-4 of paragraph [0021]), the virtual machine monitor allowing the maintenance to be performed with respect to one of the first and second instances while using the other of the first and second instances (Fig. 2, #218: maintenance is performed with respect to one of the first and second OS instances while using the other of those instances. For example, maintenance is performed on the first OS instance while using the second OS

instance: see p. 5, lines 1-5 of paragraph [0022]), wherein the virtual machine monitor configures hardware to trap instructions to create an illusion that the first instance has control of the hardware (when a trap occurs, the VMM simulates the instruction and thus creates an illusion that the OS instance has sole control of the hardware on which it runs: see p. 4, line 5 of paragraph [0018] to p. 5, lines 1-2 of paragraph [0018]).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1-43 are rejected under 35 USC § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claims 1-3, 10-23, 27-34, and 38-43 are rejected under 35 USC § 103(a) as being unpatentable over by USPN 6,698,017 (Adamovits) in view of USPN 4,347,565 (Kaneda).

Claims 4-9, 24-26, and 35-37 are rejected under 35 USC § 103(a) as being unpatentable over by USPN 6,698,017 (Adamovits) in view of USPN 4,347,565 (Kaneda) further in view of US publication number 2002/0069369 (Tremain).

VII. ARGUMENT

The rejection of claims 1 – 43 is improper, and Applicants respectfully request reversal of these rejections.

The claims do not stand or fall together. Instead, Applicants present separate arguments for various claims. Each of these arguments is separately argued below and presented with separate headings and sub-heading as required by 37 C.F.R. § 41.37(c)(1)(vii).

Overview of Claims and Primary Reference (Adamovits)

As a precursor to the arguments, Applicants provide an overview of the claims and the primary reference (Adamovits). This overview will assist in determining the scope and content of the prior art as required in Graham (see *Graham v. John Deere Co. of Kansas City*, 383 U.S. 1, 17-18 setting out an objective analysis for applying 103 rejections).

As discussed in Applicants' Background, server computers often need maintenance. If the maintenance is performed offline, then the server needs to be shut down and then re-booted. Offline maintenance leads to downtime which is expensive for businesses, especially businesses that use the servers for important or mission critical services.

The claims are directed to performing maintenance on a computer while the computer remains online. The claims recite a virtual machine monitor (VMM) that runs **two operating system (OS) instances** on the computer. One OS instances functions as a substitute for the other OS instance so maintenance can be performed with respect to the other OS instance. Specifically, the VMM configures hardware on the computer to trap instructions to create an illusion that the first OS instance has control of the hardware. The VMM simulates execution of these instructions so it appears as though the first OS instance is up and running. The specification provides numerous examples of different types of both hardware and software maintenance that can be provided with exemplary embodiments of the invention.

Adamovits teaches a systems and methods for upgrading or replacing original software with replacement or upgraded software (a process known as software migration). **Importantly, Adamovits does not use two OS instances or trapping of instructions to perform software migration.** In fact, Adamovits does not even discuss or suggest trapping instructions and using multiple OS instances, both of which have distinct meanings to one skilled in the art. Instead in Adamovits, "state information from the original software system is communicated to the replacement software system" (see Adamovits at column 2, lines 35-38). Again, virtual migration using two operating system instances is never used. By complete contrast, Adamovits teaches: "Performing a software migration on the active processing element while the original software system controls the active processing element removes the need for having a duplicate processing element to support the migration ..." (see Adamovits at column 2, lines 42-46).

Claim Rejections: 35 USC § 112

Claims 1-43 are rejected under 35 USC § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. These rejections are traversed.

Regarding claim 1, the Examiner argues that it is uncertain what the relationship is between "configuring hardware to trap instructions ... simulating trapped instructions with the virtual machine monitor" (see final OA mailed 11/16/2007). The Examiner then proceeds to ask questions with regard to what simulating an instruction has to do with performing maintenance.

Regarding claims 21 and 33, the Examiner does not understand how an illusion is created that one OS instance has control over the hardware. Applicants respectfully ask the Board to read paragraph [0018] in the original specification. This paragraph describes how instructions are trapped, and how the VMM simulates execution of the instructions to create an illusion that the OS instance in fact has sole control of the hardware.

The Examiner is not applying the proper standard to review the claims. The uncertainty and questions of the Examiner stem from a lack of understanding of operating systems, virtual machine monitors, and trapping instructions. Clearly, one skilled in the

art would have a fundamental understanding of operating systems, virtual machines, and trapping of instructions. These concepts are fundamental and basic to one of ordinary skill in this art.

Furthermore, the specification describes with words and drawings operating systems, virtual machine monitors, and trapping instructions. For brevity, Applicants will not reproduce large portions of the specification. Instead, Applicants respectfully ask the Board to read the detailed description from paragraph [0015] on p. 3 to about the end of p. 7 paragraph [0030]. This section provides an overview of how operating systems, virtual machine monitors, and trapping function together.

Further yet, the present application incorporates by reference numerous other patent applications directed to using virtual machine monitors to partition hardware (see U.S. serial number 10/676,921 filed on October 1, 2003 incorporated by reference in paragraph [0052]) or using virtual machine monitors to perform maintenance on a computer (see U.S. serial number 10/676,557 filed on October 1, 2003 incorporated by reference in paragraph [0053]). One skilled in the art would be savvy to retrieve, review, and read these other applications incorporated by reference.

In view of the clear teachings in the specification and knowledge of one skilled in the art, Applicants respectfully disagree with the Examiner's assertions and traverse the rejections, because the above-listed claim terms are not indefinite and, thus, in compliance with § 112, second paragraph. Section 112, second paragraph, requires that an applicant's claims particularly point out and distinctly claim the subject matter which an applicant regards as his invention. To satisfy this threshold, claim recitations must allow one skilled in the art to understand the bounds of the claim when read in light of the specification. *See Exxon Research and Engineering cov United States*, 60 U.S.P.Q. 2d 1272, 1276 (Fed. Cir. 2001). Thus, it is only if "a claim is *insolubly ambiguous*, and no narrowing construction can be properly adopted" can a claim be held as indefinite. *See id.* (Emphasis added). The Federal Circuit has made clear that "[i]f the meaning of a claim is discernable even though the task may be formidable and the conclusions may be one over which reasonable persons will disagree," the claim will be viewed sufficiently clear to avoid indefiniteness. *See id.*

For at least these reasons, Applicants respectfully ask the Board to reverse these rejections.

Claim Rejections: 35 USC § 103(a)

Claims 1-3, 10-23, 27-34, and 38-43 are rejected under 35 USC § 103(a) as being unpatentable over by USPN 6,698,017 (Adamovits) in view of USPN 4,347,565 (Kaneda). These rejections are traversed.

The claims recite one or more elements that are not taught or suggested in Adamovits in view of Kaneda. These missing elements show that the differences between the combined teachings in the art and the recitations in the claims are great. As such, the pending claims are not a predictable variation of the art to one of ordinary skill in the art Some examples are provided below for claims argued with separate sub-headings.

Sub-Heading: Independent Claim 1 and Its Dependent Claims

As one example, claim 1 recites running a first operating system (OS) instance on a virtual machine monitor (VMM). The claim then recites running a second OS instance on the VMM as a substitute for the first OS instance. In other words, the claim recites two operating system instances both being run on the virtual machine monitor. The second operating system instances is a substitute for the first operating system instance. The Examiner argues that this element is taught in Adamovits at column 1, line 65 to column 2, line 5 and Replacement Software System 70 of Figure 2. Applicants respectfully disagree.

The cited section of Adamovits teaches migrating control of an active processing element (i.e., a processor) from an in-service original software system to a replacement software system. Nowhere does Adamovits teach or even suggest using a virtual machine monitor that runs two operating system instances. **Adamovits does not even use two different operation system instances on a virtual machine to perform the migration.** Instead in Adamovits, "state information from the original software system is communicated to the replacement software system" (see Adamovits at column 2, lines 35-38). Again, virtual migration using two operating system instances is never used. By complete contrast, Adamovits teaches: "Performing a software migration on the active

processing element while the original software system controls the active processing element removes the need for having a duplicate processing element to support the migration ..." (see Adamovits at column 2, lines 42-46).

Furthermore, Adamovits does not teach or suggest that the replacement software system 70 of Fig. 2 is a second OS instance that substitutes for another OS instance. The Examiner is making assumptions about the teachings of Adamovits that are not taught or suggested in Adamovits. In other words, the Examiner is arguing that the original software system 50 in Adamovits and the replacement software system 70 (as shown in Fig. 2 of Adamovits) are the claim elements of first and second operating system instances that run on a virtual machine. This reading of Adamovits is not accurate. The virtual machine layer is Adamovits does not run two operating system instances as recited in the claims. Instead, Adamovits explains the function of his virtual machine layer and how migration works (column 6, lines 22-41):

> [T]he virtual machine 30 provides a migration layer, which serves as a virtual interface between hardware and software elements for the software migration. When the original software system 50 or the replacement software system 70 is fully loaded, active and in control of the active processing element 10 under normal operating conditions, the virtual machine 30 is inactive or not present so that as much of the CPU time of the processor 12 as possible is available to support the operations of the software system in control of the active processing element 10.
>
> As is further described below, the original software system 50 runs and manages the active processing element 10 while the replacement software system 70 is loaded into the second memory partition 20 (FIG. 1), provisioning information 48 (including provisioning data and provisioning code) is transferred to the replacement software system 70 (in step 116 of FIG. 6), and dynamic state information 80 for the replacement software system 70 is synchronized with dynamic state information 60 of the original software system 50 (in step 123 of FIG. 9).

Nowhere does Adamovits teach that his virtual machine layer runs two operating system instances and that one of these OS instances is a substitute for the other OS instance.

The differences between the claims and the teachings in the Adamovits and Kaneda are great since the references fail to teach or suggest all of the claim elements. As such, the pending claims are not a predictable variation of the art to one of ordinary skill in the art. For at least these reasons, the claims are allowable over the art of record.

As another example, claim 1 recites the following:

> configuring hardware to trap instructions executed by the first operating system instance;
>
> simulating trapped instructions with the virtual machine monitor.

In other words, claim 1 recites trapping instructions and then simulating the trapped instructions. The Examiner admits that Adamovits does not teach these two elements (see Final OA mailed 11/16/2007 at p. 4). Applicants agree with this admission. The Examiner, however, attempts to cure these deficiencies with Kaneda at column 2, lines 10-23 and column 4, lines 1-13. Applicants respectfully disagree.

Column 2, lines 10-23 in Kaneda discusses several functions of a virtual machine monitor (VMM). Namely, the VMM (1) controls user programs, (2) simulates privileged instructions, and (3) receives and processes interrupts. Notice that nowhere does this section of Kaneda teach or even suggest trapping instructions and then simulating the trapped instructions.

Column 4, lines 1-13 in Kaneda discusses an address control system for software simulation. A processor simulates the execution of an instruction of a program. Notice that nowhere does this section of Kaneda teach or even suggest trapping instructions and then simulating the trapped instructions.

The differences between the claims and the teachings in the Adamovits and Kaneda are great since the references fail to teach or suggest all of the claim elements. As such, the pending claims are not a predictable variation of the art to one of ordinary skill in the art. For at least these reasons, the claims are allowable over the art of record.

Sub-Heading: Independent Claim 21 and 33 and Their Dependent Claims

As one example, claims 21 and 33 recites "wherein the virtual machine monitor configures hardware to trap privileged instructions to create an illusion that the first instance has control of the hardware."

Nowhere does Adamovits teach or even suggest that the virtual machine monitor configures hardware to trap privileged instructions. Further, Adamovits does not teach or even suggest that instructions are trapped to create an illusion that an instance of an operating system has control of hardware.

Kaneda does teach that a VMM simulates privileged instructions. Kaneda, however, does not teach that these privileged instructions are trapped. Kaneda also does not teach that these instructions are trapped to "create an illusion that the first instance has control of the hardware."

The Examiner has not cited a section in Adamovits or Kaneda for allegedly teaching this recitation. Traps or trapping instructions is a specific way to handle computer operations. Further, instructions can be simulated without being first trapped. This methodology is taught in Kaneda: Privileged instructions are simulated, but they are not first trapped. Furthermore, although Kaneda does simulate instructions, they are not simulated to create an illusion that an OS instance has control of hardware.

The differences between the claims and the teachings in the Adamovits and Kaneda are great since the references fail to teach or suggest all of the claim elements. As such, the pending claims are not a predictable variation of the art to one of ordinary skill in the art. For at least these reasons, the claims are allowable over the art of record.

As another example, claims 21 and 33 recites a VMM that runs two OS instances. The second OS instance is rung as a substitute for the first OS instance when maintenance is being performed on the first OS instance. The Examiner argues that this element is taught in Adamovits at column 1, line 65 to column 2, line 5 and Replacement Software System 70 of Figure 2. Applicants respectfully disagree.

The cited section of Adamovits teaches migrating control of an active processing element (i.e., a processor) from an in-service original software system to a replacement software system. Nowhere does Adamovits teach or even suggest using a virtual machine

monitor that runs two operating system instances. **Adamovits does not even use two different operation system instances on a virtual machine to perform the migration.** Instead in Adamovits, "state information from the original software system is communicated to the replacement software system" (see Adamovits at column 2, lines 35-38). Again, virtual migration using two operating system instances is never used. By complete contrast, Adamovits teaches: "Performing a software migration on the active processing element while the original software system controls the active processing element removes the need for having a duplicate processing element to support the migration ..." (see Adamovits at column 2, lines 42-46).

Furthermore, Adamovits does not teach or suggest that the replacement software system 70 of Fig. 2 is a second OS instance that substitutes for another OS instance. The Examiner is making assumptions about the teachings of Adamovits that are not taught or suggested in Adamovits. In other words, the Examiner is arguing that the original software system 50 in Adamovits and the replacement software system 70 (as shown in Fig. 2 of Adamovits) are the claim elements of first and second operating system instances that run on a virtual machine. This reading of Adamovits is not accurate. The virtual machine layer is Adamovits does not run two operating system instances as recited in the claims. Instead, Adamovits explains the function of his virtual machine layer and how migration works (column 6, lines 22-41):

> [T]he virtual machine 30 provides a migration layer, which serves as a virtual interface between hardware and software elements for the software migration. When the original software system 50 or the replacement software system 70 is fully loaded, active and in control of the active processing element 10 under normal operating conditions, the virtual machine 30 is inactive or not present so that as much of the CPU time of the processor 12 as possible is available to support the operations of the software system in control of the active processing element 10.
>
> As is further described below, the original software system 50 runs and manages the active processing element 10 while the replacement software

> system 70 is loaded into the second memory partition 20 (FIG. 1), provisioning information 48 (including provisioning data and provisioning code) is transferred to the replacement software system 70 (in step 116 of FIG. 6), and dynamic state information 80 for the replacement software system 70 is synchronized with dynamic state information 60 of the original software system 50 (in step 123 of FIG. 9).

Nowhere does Adamovits teach that his virtual machine layer runs two operating system instances and that one of these OS instances is a substitute for the other OS instance.

The differences between the claims and the teachings in the Adamovits and Kaneda are great since the references fail to teach or suggest all of the claim elements. As such, the pending claims are not a predictable variation of the art to one of ordinary skill in the art. For at least these reasons, the claims are allowable over the art of record.

Sub-Heading: Dependent Claims 2 and 22

Dependent claims 2 and 22 recite migrating an application from a first OS instance to a second OS instance and then using the application on the second OS instance. This claim element is very different than the teachings in Adamovits.

In Adamovits, original software is replaced with replacement software. To perform this replacement, control of a processor is migrated from the original software system to the replacement software system. This teaching is quite different than the claim elements. Specifically, claims 2 and 22 recite migrating the application from one OS instance to another OS instance. In Adamovits, an application is not being migrated between two OS instances and used on the second OS instance. Adamovits does not even use two OS instances. Instead, Adamovits is performing a software migration on the processor.

Claim Rejections: 35 USC § 103(a)

Claims 4-9, 24-26, and 35-37 are rejected under 35 USC § 103(a) as being unpatentable over by USPN 6,698,017 (Adamovits) in view of USPN 4,347,565

(Kaneda) further in view of US publication number 2002/0069369 (Tremain). These rejections are traversed.

As noted above, Adamovits and Kaneda do not teach or even suggest all the elements of independent claims 1, 21, and 33. Tremain does not cure these deficiencies. Dependent claims 4-9, 24-26, and 35-37 are allowed for at least these reasons.

CONCLUSION

In view of the above, Applicants respectfully request the Board of Appeals to reverse the Examiner's rejection of all pending claims.

Any inquiry regarding this Amendment and Response should be directed to Philip S. Lyren at Telephone No. 832-236-5529. In addition, all correspondence should continue to be directed to the following address:

Hewlett-Packard Company
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400

Respectfully submitted,

/Philip S. Lyren #40,709/

Philip S. Lyren
Reg. No. 40,709
Ph: 832-236-5529

VIII. Claims Appendix

1. A method of performing online computer maintenance on at least one node, the method comprising:

running a virtual machine monitor;

running a first operating system instance on the virtual machine monitor;

configuring hardware to trap instructions executed by the first operating system instance;

simulating trapped instructions with the virtual machine monitor;

running a second operating system instance on the virtual machine monitor as a substitute for the first operating system instance; and

performing the maintenance with respect to one of the first and second operating system instances while using the other of the first and second operating system instances.

2. The method of claim 1, wherein the second operating system instance is run as a substitute by migrating at least one application from the first operating system instance to the second operating system instance, and using the migrated applications on the second operating system instance.

3. The method of claim 1, further comprising shutting down the first operating system instance after the second operating system instance has been run as a substitute for the first operating system instance.

4. The method of claim 1, wherein the maintenance includes hardware servicing; and

wherein the second operating system instance is run without a dependency on hardware to be serviced.

5. The method of claim 4, wherein the servicing includes removing hardware from the at least one node.

6. The method of claim 4, wherein the virtual machine monitor is used to hide hardware to be serviced from the second operating system instance during bootup of the second operating system instance.

7. The method of claim 4, wherein the virtual machine monitor releases its own dependencies on hardware prior to removal.

8. The method of claim 1, wherein the maintenance includes adding the hardware; wherein the virtual machine monitor discovers the hardware; and wherein the virtual machine monitor shields the first operating system instance from the hardware as the hardware is being added.

9. The method of claim 8, wherein the hardware is added before the second operating system instance is booted; and wherein the second operating system instance is allowed during bootup to see the hardware.

10. The method of claim 1, wherein applications running on the first operating system

instance are migrated to the second operating system instance; and wherein software maintenance is performed.

11. The method of claim 10, further comprising shutting down one of the first and second operating system instances after the applications have been migrated.

12. The method of claim 1, wherein the second operating system instance is an upgraded operating system; and wherein applications running on the first operating system instance are migrated to the second operating system instance.

13. The method of claim 1, wherein the maintenance includes modifying the second operating system instance; and wherein the method further includes migrating applications running on the first operating system instance to the second operating system instance.

14. The method of claim 1, further comprising migrating applications from the first operating system instance to the second operating system instance before the maintenance is performed; migrating the applications from the second operating system instance back to the first operating system instance after the maintenance has been performed; and shutting down the second operating system instance following the application migration to the first operating system instance.

15. The method of claim 1, wherein a first application instance is running on the first

operating system instance before the maintenance is performed; and wherein the maintenance includes running a second application instance on the second operating system instance, modifying the second application instance, and cutting over from the first application instance to the modified second application instance.

16. The method of claim 1, wherein the virtual machine monitor allows at least one of the first and second operating system operating system instances to have direct control over at least one of a processing unit, memory and I/O of the at least one node.

17. The method of claim 1, wherein the first operating system instance is booted prior to running the virtual machine monitor; and wherein the virtual machine monitor is interposed beneath the first operating system instance when maintenance is to be performed.

18. The method of claim 1, wherein at least one of a processing unit, memory and I/O is devirtualized after the maintenance has been performed.

19. The method of claim 1, wherein a single processor is used to run the virtual machine monitor and the first and second operating system instances.

20. The method of claim 1, wherein a single node is used to run the virtual machine monitor and the first and second operating system instances.

21. A node comprising a processing unit and memory for the processing unit, the memory encoded with a virtual machine monitor and an operating system, the virtual machine monitor running a second instance of the operating system when a first instance of the operating system is already running in the node, the second instance being run when maintenance is to be performed, the second instance being a substitute for the first instance, the virtual machine monitor allowing the maintenance to be performed with respect to one of the instances while using the other of the instances wherein the virtual machine monitor configures hardware to trap privileged instructions to create an illusion that the first instance has control of the hardware.

22. The node of claim 21, wherein the second instance is run as a substitute by migrating at least one application from the first instance to the second instance, and using the at least one application on the second instance.

23. The node of claim 21, wherein the virtual machine monitor shuts down the first instance after the second instance has been run as a substitute for the first instance.

24. The node of claim 21, wherein the virtual machine monitor is used to hide hardware to be serviced from the second instance during bootup of the second instance.

25. The node of claim 21, wherein the maintenance includes adding hardware; wherein the virtual machine monitor discovers the added hardware; and wherein the virtual machine monitor shields the first instance from the added hardware as the added

hardware is being added.

26. The node of claim 25, wherein the added hardware is added before the second instance is booted; and wherein during bootup the virtual machine monitor allows second instance to see the added hardware.

27. The node of claim 21, wherein applications running on the first instance are migrated to the second instance; and wherein software maintenance is performed.

28. The node of claim 21, wherein the maintenance includes modifying the second instance; and wherein applications running on the first instance are migrated to the second instance.

29. The node of claim 21, wherein applications are migrated from the first instance to the second instance before the maintenance is performed; the applications are migrated from the second instance back to the first instance after the maintenance has been performed; and wherein the virtual machine monitor shuts down the second instance after the applications are migrated to the first instance.

30. The node of claim 21, wherein the virtual machine monitor allows at least one of the first and second instances to have direct control over at least one of a processing unit, memory and I/O of the at least one node.

31. The node of claim 21, wherein the first instance is booted prior to running the virtual machine monitor; and wherein the virtual machine monitor is interposed beneath the first instance when maintenance is to be performed.

32. The node of claim 21, wherein the virtual machine monitor devirtualizes at least one of a processing unit, memory and I/O after the maintenance has been performed.

33. An article for a processing unit of a node, the article comprising computer memory encoded with a virtual machine monitor for running first and second instances of an operating system, the second instance being run when maintenance on the node is to be performed, the second instance being a substitute for the first instance, the virtual machine monitor allowing the maintenance to be performed with respect to one of the first and second instances while using the other of the first and second instances, wherein the virtual machine monitor configures hardware to trap instructions to create an illusion that the first instance has control of the hardware.

34. The article of claim 33, wherein the virtual machine monitor shuts down the first instance after the second instance has been run as a substitute for the first instance.

35. The article of claim 33, wherein the virtual machine monitor is used to hide hardware to be serviced from the second instance during bootup of the second instance.

36. The article of claim 33, wherein the maintenance includes adding hardware; wherein

the virtual machine monitor discovers the added hardware; and wherein the virtual machine monitor shields the first instance from the added hardware as the added hardware is being added.

37. The article of claim 36, wherein the added hardware is added before the second instance is booted; and wherein during bootup the virtual machine monitor allows the second instance to see the added hardware.

38. The article of claim 33, wherein the virtual machine monitor causes applications running on the first instance to be migrated to the second instance in order to perform software maintenance.

39. The article of claim 33, wherein the maintenance includes modifying the second instance; and wherein the virtual machine monitor causes applications running on the first instance to be migrated to the second instance.

40. The article of claim 33, wherein the virtual machine monitor causes applications to be migrated from the first instance to the second instance before the maintenance is performed; wherein the virtual machine monitor causes the applications to be migrated from the second instance back to the first instance after the maintenance has been performed; and wherein the virtual machine monitor shuts down the second instance after the applications are migrated to the first instance.

41. The article of claim 33, wherein the virtual machine monitor allows at least one of the first and second instances to have direct control over at least one of a processing unit, memory and I/O of the at least one node.

42. The article of claim 33, wherein first instance is booted prior to running the virtual machine monitor; and wherein the virtual machine monitor is interposed beneath the first instance when maintenance is to be performed.

43. The article of claim 33, wherein the virtual machine monitor can devirtualize at least one of a processing unit, memory and I/O after the maintenance has been performed.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

None.